

EPG-II

EDIT PROGRAM GENERATOR SYSTEM FOR BUSINESS APPLICATIONS

GENERAL INFORMATION MANUAL

COMPUTER SCIENCES CORPORATION

EPG- II

Edit Program Generator System For Business Applications

General Information Manual

August 1969
Second Printing
February 1970

APPLICATIONS SYSTEMS DIVISION
COMPUTER SCIENCES CORPORATION

650 North Sepulveda Boulevard—El Segundo, California 90245

Offices and Facilities in Major Cities Throughout the United States, Canada, Europe, and the Pacific Area



EPG-II: Edit Program Generator System for Business Applications

CONTENTS

Introductory	v	Who Benefits From EPG-II?	5
Why EPG-II?	1	For Senior Management	5
What is EPG-II?	3	For EDP Department Management	5
Error Detection	4	EPG-II In Use	6
Self-Explanatory Edits	4	How Does EPG-II Work?	7
Edit Standardization	4	Code Preparation	7
Debugging Aid	4	Special Features	8
Minimal Core Environment	4	Examples	10
		About Computer Sciences Corporation	17

INTRODUCTORY

An "ideal" computer program edit system has been defined as one which meets these criteria:

- Detects input errors
- Produces clear, self-explanatory edit messages
- Standardizes edit procedures
- Aids in system or program debugging
- Functions in a minimal core environment

EPG-II -- Edit Program Generator System for Business Applications -- is a tested and proven proprietary product of Computer Sciences Corporation, created and designed to bring reality in line with the ideal edit.

1

WHY EPG-II?

Most data processing professionals readily agree on the need for input or "front-end" editing of computer software systems and the programs which comprise systems. Indeed, many software technologists would find it difficult to imagine a data processing application without some sort of editing function.

Without a competent edit, two results are almost always produced: processing programs function incorrectly, if at all, and output ranges from the unreliable to the useless. A common dictum among professionals in the industry is "garbage in/garbage out", clearly indicating the wide spread recognition of the need for dependable and thorough input editing.

The unfortunate reality is that editing programs, in many situations, fall far short of meeting the criteria

— detecting input errors, producing clear and self-explanatory edit messages, standardizing edit procedures, and aiding in system or program debugging, while operating within a minimal core requirement — for the "ideal" edit.

The consequences of this unfortunate reality are ultimate-user dissatisfaction with the output and unnecessarily high programming system costs.

Computer Sciences Corporation (CSC) presents as its answer to these problems

EPG-II,

the Company's Edit Program Generator System for Business Applications, as an efficient means for implementing input edits which meet the criteria for reliable systems performance. EPG-II has been tested and proven in user facilities across the United States and is available for immediate installation.

2

WHAT IS EPG-II?

EPG-II is an edit program generator designed to meet the requirements of virtually any edit situation encountered in normal business data processing applications. To perform its program generation function, EPG-II employs a specialized edit language of some 60 mnemonics which provide the user with full language capability.

An analyst or programmer is required only to write the appropriate edit mnemonics for a proposed program. EPG-II assembles or compiles the statements to generate the edit program, which in turn becomes the input or "front-end" edit to the application program.

As EPG-II reads its coded edit language input and translates that input into executable computer instructions, it performs these two meaningful functions:

- 1) EPG-II diagnoses the code for validity of reference to the language specifications and capabilities, simultaneously producing a print-out of diagnostic messages explaining any errors in the input.
- 2) EPG-II prints English language documentation with reference to any edit language statement in question, together with a full explanation of the nature of the statement and the action required to satisfy the conditions contained in the statement.

The end result is an error-free program, produced in far less time than is required for individual edit program writing, and with many fewer computer passes required to prove the validity of the edit and

its system or individual program application. EPG-II functional characteristics, previously referenced, are discussed individually in the paragraphs which follow.

Error Detection

EPG-II provides the capability to detect errors by editing the validity of field contents, the validity of card hierarchial relationships, conditional requirements within cards, cross-foot totals within cards, and by verifying batch and grand totals to external controls.

Self-Explanatory Edits

The error message capabilities of EPG-II have been refined through exposure to a variety of users. These capabilities have demonstrated themselves to be self-explanatory, providing the technologist with ready understanding of a detected error and the required corrective action.

Edit Standardization

EPG-II, by the very nature of its standardized approach to edits and edit messages, provides an effective means for standardization of all edit procedures within an EDP department or installation; this condition prevails as personnel changes occur, or as higher priority assignments necessitate the employment on a project of personnel not familiar with the development of the subject application.

Debugging Aid

EPG-II permits test data for a system to be edited precisely as live input data is to be edited. At the same time, EPG-II enables test data to be created quickly with the assurance that the data is "clean".

Minimal Core Environment

The system operates in a minimal core environment; most EPG-II applications will execute in less than 20K of core memory.

3

Who Benefits From EPG-II?

From the viewpoint of corporation or institution management, valid and meaningful outputs produced promptly on a cost-effective basis are the criteria for evaluating computer services. From the viewpoint of data processing professionals, fast and accurate responses to management requirements, produced on the basis of lowered unit costs, are the criteria for efficient system performance. EPG-II is a means for accomplishing the objectives of both senior management and departmental management in producing well-structured reports, accurately and on a timely basis, and at minimum implementation cost.

For Senior Management

EPG-II saves time in terms of responsiveness to requests from managers for vital information on a

quick turn-around basis. This is accomplished in a number of ways, particularly with EPG-II's reduced requirement for time required to write and debug programs. Edit operations are reduced to minutes for simple programs, a few hours for more complex programs, compared with the man-days which are required in many data processing installations. Since management can depend on those reports, the effects of enlightened executive decisions, based upon the more accurate and timely information EPG-II makes possible, are then reflected in operating results.

For EDP Department Management

The problem which persists in most EDP installations, the lack of sufficient trained personnel, is minimized with EPG-II. The system's standardization

features enable personnel to be moved from application to application without lengthy orientation in the edit status of a new application. At the same time, quick and comprehensive error detection and identification enable final implementation of an application in far less time than a user normally experiences without EPG-II. Less editing and test evaluation time consumed produces greater flexibility within limited budgets.

EPG-II In Use

As an edit programming language, EGP-II is logical, straightforward, and consistent. The instruction set and coding conventions are directed toward ease of use. EPG-II dictates that the edit be thoroughly defined when it is programmed - the unanticipated is not overlooked. Therefore, relatively inexperienced programmers can become proficient in the use of the language with less than one day's training.

Since EPG-II is an edit program generator, diagnostics are edit-oriented. Code processed by EPG-II is analyzed for validity, and any coding errors are noted along with self-explanatory diagnostic messages indicating the nature of the individual error.

EPG-II is self-documenting. When input is processed through the EPG-II system to create the edit program, a complete set of documentation for the program also is created. Each statement is followed by a narrative description of the statement, describing in English the exact actions to be taken by the programmer in accordance with that statement. No manually-written documentation is required. Further, EPG-II edit programs have been proved to be easier to write than narrative edit specifications. The self-documenting EPG-II program, written in place of edit specifications, also eliminates the possibility of misinterpretation of written specifications.

EPG-II audits master files and assures the validity of those files. As an auditor, the system provides a means for correcting invalid data, and previously-programmed weak edits can be identified and corrected or rewritten. When master files contain inaccurate or improper data, EPG-II can be employed to improve the condition of those existing files. The accuracy and completeness of the EPG-II edit requirements applied to master files cause the professional's confidence level in the files to be raised significantly.

Most EPG-II users have found that the system helps to create a positive atmosphere in which new systems and approaches can be undertaken by the Data Processing Department in behalf of its company users.

The widely-recognized problem of communications between and among departments is minimized, insofar as report requirements and output efficacy are concerned, when EPG-II is employed. Error communication messages to the user are clearly understood, and the validated edit removes the possibility of incomplete edits which otherwise clouds the issue.

It is in the area of cost reduction, apart from the system's utility, that EPG-II perhaps produces its most beneficial effect. EPG-II coding times are less than times required to prepare final specifications for the coding of edit problems, or roughly equal to the time required to prepare a File Definition in the Data Division of a COBOL program; thus, all of the coding time and attendant costs normally expended preparing the actual logic of the edit in a COBOL-type program are completely eliminated. Most EPG-II programs can be compiled and debugged in three compilation passes, effecting a further saving. EPG-II can be used for the preparation of test data, as all systems testing can be done with input created by the edit program, edited to the true specifications of the system, and eliminating thereby the requirement for programmers to develop their own test data for individual segments of a system. Control desk and maintenance costs are similarly reduced.

4

HOW DOES EPG-II WORK?

EPG-II generates edit programs for use with the IBM System/360-30 or larger, operating in either DOS or OS environments; tape or disk systems are accommodated. Partition requirements are minimal, depending upon a particular hardware configuration.

Edit programs generated by EPG-II read input transactions, verify key numbers to external files or tables, edit the format and content of input records to the edit specifications, make relational tests between input records, accumulate batch and grand total balances and check these to batch control inputs, format and write output files for subsequent processing, and prepare printed processing logs. These processing logs constitute journals of all batches submitted for a particular run, with both the specified and computed totals for each batch. Within each batch, detected errors are printed along with an image

of the error record. When an error is detected, an error message is printed, and the editing continues. Thus, one input record could produce several error messages.

Code Preparation

To write an EPG-II program, three basic types of code are prepared - control packets, edit packets, and logic packets. Each packet is looked upon as a separate routine, designed to perform a specific function within the generated edit program.

Control Packets — are used to define the identification of the program, input files, output files, record types within input files, external files, and external tables. Based upon the contents of these control

packets, EPG-II creates the necessary code to define, read, and write files and to accumulate necessary batch controls.

Edit Packets – are created for each input record type. These packets define all of the edit tests to be performed on a given record type as well as the formatting criteria for each output record. EPG-II then generates the code necessary to perform the actual editing and formatting functions.

Logic Packets – are used as subroutines within edit packets to define logical relationships among various fields on a given input record. These relationships may take one of the following forms:

- 1) if a specific condition is met in one field, then some other condition must also be met in another field;
- 2) at least one of several conditions must be met in a particular field or in several fields;
- 3) all of a set of conditions must be met in one or more fields, or
- 4) some combination of the first three.

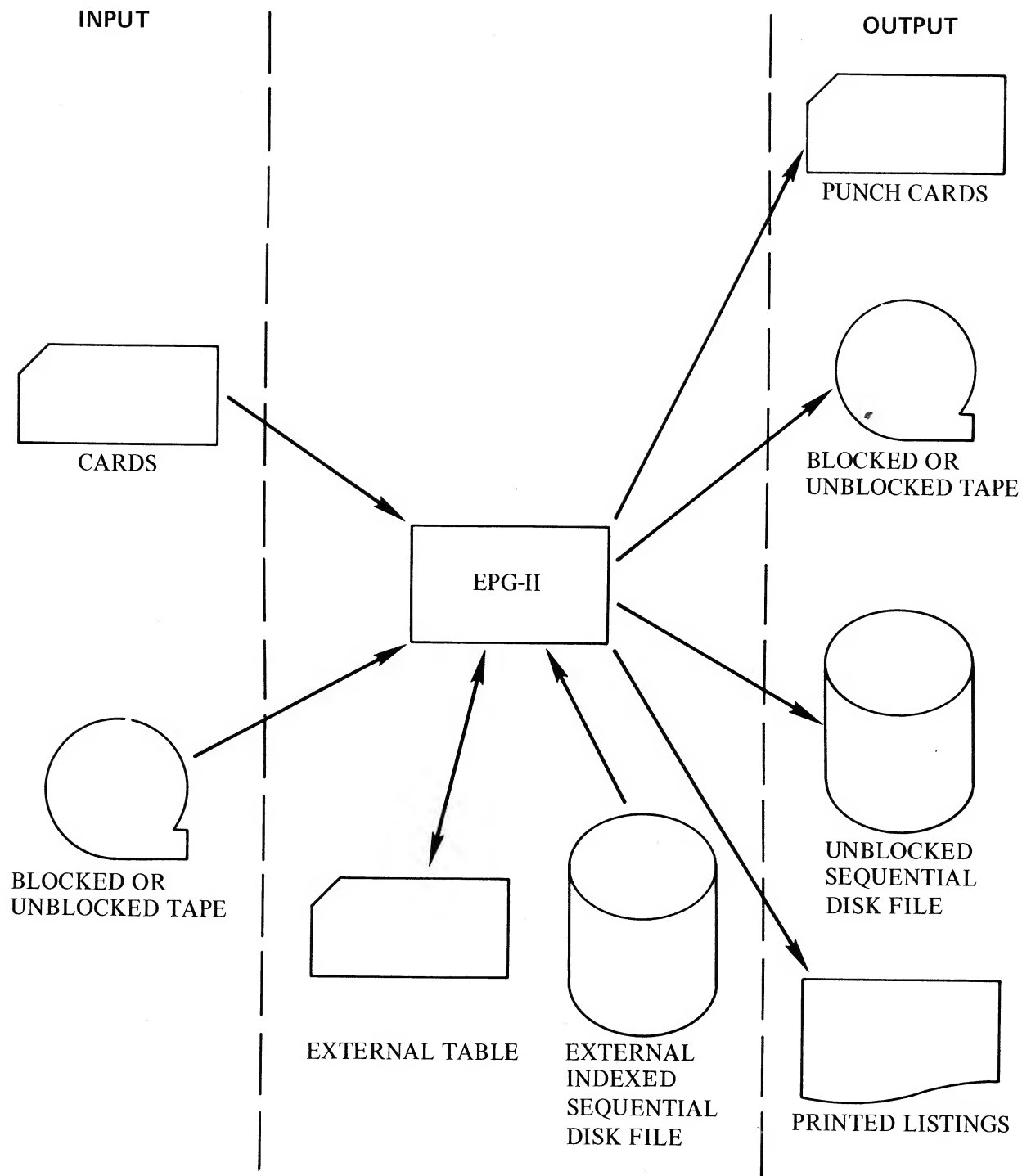
Logical statements are provided to specify the editing criteria for such logical relationships testing and are used in conjunction with the normal editing statements provided in EPG-II.

Special Features

In addition to the basic packet types designed to accomplish equally basic editing functions within EPG-II, three additional capabilities are provided to handle additional types of editing circumstances frequently encountered. These situations are treated in the scratch area, in external tables, and in external files.

The scratch area is a work area which provides the user with the capability of either saving data for reference while processing subsequent records, or for building fields prior to moving to an output formatting area. The size of the scratch area may be defined by the user. EPG-II also provides a capability for first building external tables and then referencing and extracting data from the tables as an edit is performed. Indexed sequential files also may be used for reference and retrieval purposes as an edit is performed.

Figure 1 on the following page indicates the input and output devices optionally compatible with EPG-II system performance.



Examples

Comparative examples of the EPG-II programming required to edit an 80-column card input are presented in the figures which follow. The fields edited by the COBOL and EPG-II programs include those for date, product number, sales quantity, three sales areas, sales dollars, and description.

The apparently abbreviated Figure 3 actually

illustrates all of the EPG-II source statements required to produce the edit program for the specified application.

Figure 4 is the compile listing stemming from the EPG-II source statements.

The three pages comprising Figure 6 are the COBOL source statements required to perform the same edit functions performed by the EPG-II program shown in Figure 3.

ID	DATE	PROD NO.	SALES QTY	SALES AREA			SALES DOLLARS	DESCRIPTION
				1	2	3		
1	5	11	17	24	29	34	39	49 70
ID <i>Card identification</i> DATE <i>MMDDYY</i> PRODUCT NUMBER <i>AANNNN - Range 0001-5999</i> SALES QTY <i>Right justified digits</i> SALES AREAS <i>Cross foot and verify with sales qty</i> SALES DOLLARS <i>Take batch total</i> DESCRIPTION BLANK								

Figure 2. Card Input

IDENT CODE=BB,APPL='EXAMPLE 1'	EX100100
INPUT CARD=80	EX100200
OUTPT DISK=1,DKLEN=70	EX100300
COTYP, BB01, BB02	EX100400
HEADR BB01	EX100500
EDITS BB02	EX100600
DATE 5,10,6	EX100700
VALID 11,12,ALPHA	EX100800
VALID 13,16,DIGIT	EX100900
UPLIM 13,16,5999	EX101000
LOLIM 13,16,0001	EX101100
VALID 17,23,DIGIT	EX101200
BLRJD 24,28	EX101300
BLRJD 29,33	EX101400
BLRJD 34,38	EX101500
VALID 39,48,DIGIT	EX101600
VALID 49,70,ALPHA,BLANK,DIGIT	EX101700
CROSS 24,28,29,33,34,38,17,23	EX101800
BATCH 39,48,48,1	EX101900
BLANK 71,80	EX102000
MOVE 70,INPUT,1,DISKR1,1	EX102100
ENDIT BB02	EX102200
END	EX102300

Figure 3. EPG-11 Source Statements

STMT	SOURCE STATEMENT	FDDS CL3-3 PAGE 1
2	IDENT CODE=BB,APPL='EXAMPLE 1'	
8	INPUT CARD=80	EX100200
9	*,CARD INPUT, SYSUNIT IS SYS008	
10	*,RECORD LENGTH IS 80 CHARACTERS.	
11	OUTPT DISK=1,DKLEN=70	EX100300
12	*,DISK OUTPUT SPECIFIED, FILENAME IS BBDASD.	
13	*,RECORD LENGTH IS 70 CHARACTERS.	
14	*,FILENAME FOR THE RESTART FILE IS BBEPGL.	
15	*,VALID RECORD NAMES FOR MOVE STATEMENTS ARE -	
16	*, DISKR1	
17	CDTYP BB01,BB02	EX100400
18	HEADR BB01	EX100500
726	EDITS BB02	EX100600
731	DATE 5,10,6	EX100700
740	*,VERIFY THAT THE DATE IN COLS 5 THRU 10 IS VALID.	
741	VALID 11,12,ALPHA	EX100800
742	*,ALPHABETIC	
752	*,VERIFY THAT COLS 11 THRU 12 CONTAIN VALID CHARACTERS	
753	VALID 13,16,DIGIT	EX100900
754	*,NUMERIC	
764	*,VERIFY THAT COLS 13 THRU 16 CONTAIN VALID CHARACTERS	
765	UPLIM 13,16,5999	EX101000
776	*,VERIFY THAT COLS 13 THRU 16 ARE LESS THAN 5999	
777	LOLIM 13,16,0001	EX101100
788	*,VERIFY THAT COLS 13 THRU 16 ARE GREATER THAN 0001	
789	VALID 17,23,DIGIT	EX101200
790	*,NUMERIC	
800	*,VERIFY THAT COLS 17 THRU 23 CONTAIN VALID CHARACTERS	
801	BLRJD 24,28	EX101300
821	*,COLS 24 THRU 28 MUST BE BLANK OR RIGHT-JUST DIGITS	
822	BLRJD 29,33	EX101400
842	*,COLS 29 THRU 33 MUST BE BLANK OR RIGHT-JUST DIGITS	
843	BLRJD 34,38	EX101500
863	*,COLS 34 THRU 38 MUST BE BLANK OR RIGHT-JUST DIGITS	
864	VALID 39,48,DIGIT	EX101600
865	*,NUMERIC	
875	*,VERIFY THAT COLS 39 THRU 48 CONTAIN VALID CHARACTERS	
876	VALID 49,70,ALPHA,BLANK,DIGIT	EX101700
877	*,ALPHABETIC, BLANKS,NUMERIC	
889	*,VERIFY THAT COLS 49 THRU 70 CONTAIN VALID CHARACTERS	
890	CROSS 24,28,29,33,34,38,17,23	EX101800
891	*, 24-28	
896	*, 29-33	
901	*, 34-38	
906	*, 17-23, TOTAL	
917	*,VERIFY CROSS-FOOTING IN FIELDS ABOVE	
918	BATCH 39,48,48,1	EX101810
926	*,COMPUTE BATCH TOTALS ON COLS 39 THRU 48, USING	
927	*,ACCUMULATOR 1, SIGN IS IN COL 48	
972	BLANK 71,80	
980	*,TEST FOR BLANKS IN COLS 71 THRU 80	
981	MOVE 70,INPUT,1,DISKR1,1	
984	*,MOVE 70 COLS FROM INPUT,1 TO DISKR1,1.	
985	ENDIT BB02	EX102900
994	END	

Figure 4. Compile Listing

Card ID	Application Name	Card No. of Batch	EPG/II PROCESSING LOG	Set Date	PAGE	1
BBEPG	EXAMPLE 1			07/24/69		
ERROR MESSAGE	FIELD	CARD	1...5...1...5...2...5...3...5...4...5...5...5...6...5...7...5...8			
BATCH NUMBER 69						
CROSS-FOOTING	17 23	2	BB02060769XB444400004000020000100002000000005000 5 LB WIDGETS SPICED			
INVALID CHAR	13 16	5	BB02060769XB4X4400004000020000100001000000005000 5 LB WHATNOTS HINGED			
INVALID CHAR	49 70	7	BB02060769XB444400004000020000100001000000005000 5 LB WHATNOTS/HINGED			
NOT ALL BLANK	71 80	7	BB02060769XB444400004000020000100001000000005000 5 LB WHATNOTS/HINGED			
UNKNOWN CARD ID	01 04	9	BB21061769AA5000000010000100 000010000001000 SPOILED CATNIP			
Card Type	BB21 Invalid		ACCUMULATOR 01, COMPUTED TOTAL 0000032000+			
			END OF BATCH 69 9 CARDS, 5 ERRORS			

Figure 5. Error Processing Log

LINE	SEQ.	SOURCE STATEMENT	PAGE	2
58	004010	FD LISTING	EDIT0001	
59	004020	RECORD CONTAINS 133 CHARACTERS	EDIT0001	
60	004030	LABEL RECORDS ARE OMITTED	EDIT0001	
61	004040	RECORDING MODE IS F	EDIT0001	
62	004050	DATA RECORD IS PRINT-LINE.	EDIT0001	
63	004060	01 PRINT-LINE.	EDIT0001	
64	004070	03 CONTROL-CODE PICTURE X.	EDIT0001	
65	004080	03 PRINT-IMAGE PICTURE X(80).	EDIT0006	
66	004082	03 ERROR-CODES.	EDIT0006	
S 67	003060	05 ERROR-1 PICTURE X(01).	EDIT0001	
68	003070	05 ERROR-2 PICTURE X(01).	EDIT0001	
69	003080	05 ERROR-3 PICTURE X(01).	EDIT0001	
70	003090	05 ERROR-4 PICTURE X(01).	EDIT0001	
71	003100	05 ERROR-5 PICTURE X(01).	EDIT0001	
72	003110	05 ERROR-6 PICTURE X(01).	EDIT0001	
73	003120	05 ERROR-7 PICTURE X(01).	EDIT0001	
74	003130	05 ERROR-8 PICTURE X(01).	EDIT0001	
75	003140	05 ERROR-9 PICTURE X(01).	EDIT0001	
76	003150	05 ERROR-0 PICTURE X(01).	EDIT0002	
77	004089	03 FILLER PICTURE X(42).	EDIT0006	
78	004090	WORKING-STORAGE SECTION.	EDIT0001	
79	004100	77 INN PICTURE S9(03) VALUE +0 COMPUTATIONAL-3.	EDIT0003	
80	004110	77 OUT PICTURE S9(03) VALUE +0 COMPUTATIONAL-3.	EDIT0003	
81	004120	77 BATCH-ACCUM PICTURE S9(12) VALUE +0 COMPUTATIONAL-3.	EDIT0001	
82	004130	77 BATCH-TOTL-HOLD PICTURE X(10).	EDIT0006	
83	004140	77 BATCH-HOLD PICTURE X(02).	EDIT0001	
84	004150	77 EDIT-SWITCH PICTURE X(01) VALUE SPACE.	EDIT0001	
85	004160	77 SALES-ACCUM PICTURE S9(12) VALUE +0 COMPUTATIONAL-3.	EDIT0001	
86	004170	77 CROSS-TOTAL PICTURE S9(12) VALUE +0 COMPUTATIONAL-3.	EDIT0001	
87	004180	01 CARD-TABLE.	EDIT0003	
88	004190	03 HOLD-AREA PICTURE X(80) OCCURS 100 TIMES.	EDIT0003	
89	005010	PROCEDURE DIVISION.	EDIT0001	
90	005020	001-START.	EDIT0001	
91	005030	OPEN INPUT CARD-IN.	EDIT0001	
92	005040	OPEN OUTPUT DISK-OUT, LISTING.	EDIT0001	
93	005060	003-READ-INPUT.	EDIT0001	
94	005070	READ CARD-IN AT END GO TO 017-END-OF-JOB.	EDIT0001	
95	005150	MOVE SPACES TO PRINT-LINE.	EDIT0006	
S 96	005080	IF BATCH-ID IS EQUAL TO 'BB01' GO TO 005-BATCH-PROCESS.	EDIT0001	
97	005090	PERFORM 013-CARD-EDIT THRU 015-EDIT-END.	EDIT0001	
98	005100	IF INN = 100 MOVE 'J' TO ERROR-0.	EDIT0006	
99	005105	IF ERROR-CODES NOT = SPACES GO TO 004-PRINT-CARD-IMAGE.	EDIT0006	
100	005110	ADD SALES-DOLLARS TO SALES-ACCUM.	EDIT0006	
101	005120	ADD 1 TO INN	EDIT0002	
102	005130	MOVE SALES TO HOLD-AREA (INN).	EDIT0003	
103	005140	004-PRINT-CARD-IMAGE.	EDIT0002	
104	005160	MOVE SALES TO PRINT-IMAGE.	EDIT0001	
105	005170	MOVE '1' TO CONTROL-CODE.	EDIT0001	
106	005180	WRITE PRINT-LINE AFTER 1.	EDIT0001	
107	005190	GO TO 003-READ-INPUT.	EDIT0001	
108	006010	005-BATCH-PROCESS.	EDIT0001	
109	006015	IF BATCH-TOTL-HOLD = SPACES GO TO 007-BATCH-OK.	EDIT0006	
110	006020	IF SALES-ACCUM IS EQUAL TO BATCH-ACCUM	EDIT0006	
111	006030	GO TO 007-BATCH-OK.	EDIT0001	
112	006040	DISPLAY 'BATCH DELETED . . . OUT OF BALANCE' UPON CONSOLE.	EDIT0001	
113	006050	DISPLAY 'BATCH NUMBER . . . ' BATCH-HOLD UPON CONSOLE.	EDIT0001	
114	006060	GO TO 011-RE-INITIALIZE.	EDIT0001	

Figure 6. COBOL Source Statements, 2 of 3

LINE	SEQ.	SOURCE STATEMENT	PAGE	3
115	006070	007-BATCH-OK.	EDIT0001	
116	006073	MOVE BATCH-TOTAL TO BATCH-TOTL-HOLD.	EDIT0007	
117	006075	IF BATCH-TOTAL NOT NUMERIC MOVE ZEROS TO BATCH-TOTL-HOLD	EDIT0005	
118	006095	DISPLAY 'BATCH EDIT ERROR' UPON CONSOLE.	EDIT0006	
119	006098	IF BATCH-TOTL-HOLD NOT = SPACES MOVE BATCH-TOTAL TO	EDIT0007	
120	006099	BATCH-ACCUM.	EDIT0007	
121	006100	MOVE BATCH-NUMBER TO BATCH-HOLD.	EDIT0001	
122	006110	009-WRITE-DISK.	EDIT0001	
123	006120	ADD 1 TO OUT.	EDIT0001	
124	006130	IF OUT GREATER THAN INN GO TO 011-RE-INITIALIZE.	EDIT0003	
125	006140	WRITE SALES-FILE FROM HOLD-AREA (OUT).	EDIT0003	
126	006150	GO TO 009-WRITE-DISK.	EDIT0001	
127	007010	011-RE-INITIALIZE.	EDIT0001	
128	007020	MOVE ZEROS TO INN, OUT, BATCH-ACCUM.	EDIT0002	
129	007050	GO TO 003-READ-INPUT.	EDIT0001	
130	007060	013-CARD-EDIT..	EDIT0001	
131	007070	IF REMAINDER IS NOT EQUAL TO SPACES,	EDIT0001	
132	007080	MOVE SPACES TO REMAINDER MOVE 'I' TO ERROR-9.	EDIT0001	
133	007090	IF CARD-TRANS IS EQUAL TO 'BB01' GO TO 014-CHECK-DATE.	EDIT0004	
134	007095	IF CARD-TRANS IS EQUAL TO 'BB02' GO TO 014-CHECK-DATE.	EDIT0004	
135	007100	MOVE 'A' TO ERROR-1. GO TO 015-EDIT-END.	EDIT0004	
136	007105	014-CHECK-DATE.	EDIT0004	
137	007110	IF MONTH IS LESS THAN 01 MOVE 'B' TO ERROR-2.	EDIT0005	
138	007120	IF MONTH IS GREATER THAN 12 MOVE 'B' TO ERROR-2.	EDIT0005	
139	007130	IF DAY IS LESS THAN 01 MOVE 'B' TO ERROR-2.	EDIT0005	
140	007140	IF DAY IS GREATER THAN 31 MOVE 'B' TO ERROR-2.	EDIT0005	
141	007150	IF YEAR IS NOT NUMERIC MOVE 'B' TO ERROR-2.	EDIT0005	
142	007160	IF YEAR IS GREATER THAN 69 MOVE 'B' TO ERROR-2.	EDIT0005	
143	008010	IF PRODUCT-ALPHA IS NOT ALPHABETIC MOVE 'C' TO ERROR-3.	EDIT0001	
144	008015	IF PRODUCT-NUM IS NOT NUMERIC MOVE 'C' TO ERROR-3.	EDIT0001	
145	008020	IF PRODUCT-NUM IS LESS THAN 0001 MOVE 'C' TO ERROR-3.	EDIT0001	
146	008030	IF PRODUCT-NUM IS GREATER THAN 5999 MOVE 'C' TO ERROR-3.	EDIT0001	
147	008070	IF SALES-QUANTITY IS NOT NUMERIC MOVE 'D' TO ERROR-4.	EDIT0001	
148	008080	EXAMINE AREA-1-A REPLACING LEADING SPACES BY ZEROS.	EDIT0004	
149	008090	EXAMINE AREA-2-A REPLACING LEADING SPACES BY ZEROS.	EDIT0004	
150	008100	EXAMINE AREA-3-A REPLACING LEADING SPACES BY ZEROS.	EDIT0004	
151	008110	IF AREA-1 IS NOT NUMERIC MOVE 'E' TO ERROR-5	EDIT0005	
152	008115	GO TO 015-EDIT-END.	EDIT0005	
153	008120	IF AREA-2 IS NOT NUMERIC MOVE 'E' TO ERROR-5	EDIT0005	
154	008125	GO TO 015-EDIT-END.	EDIT0005	
155	008130	IF AREA-3 IS NOT NUMERIC MOVE 'E' TO ERROR-5	EDIT0005	
156	008135	GO TO 015-EDIT-END.	EDIT0005	
157	008140	MOVE ZEROS TO CROSS-TOTAL.	EDIT0001	
158	008150	ADD AREA-1 TO CROSS-TOTAL.	EDIT0001	
159	008160	ADD AREA-2 TO CROSS-TOTAL.	EDIT0001	
160	008170	ADD AREA-3 TO CROSS-TOTAL.	EDIT0001	
161	008180	IF CROSS-TOTAL IS NOT EQUAL TO SALES-QUANTITY,	EDIT0001	
162	008190	MOVE 'F' TO ERROR-6.	EDIT0001	
163	009010	IF SALES-DOLLARS IS NOT NUMERIC MOVE 'G' TO ERROR-7.	EDIT0001	
164	009050	015-EDIT-END.	EDIT0001	
165	009060	EXIT.	EDIT0001	
166	009070	017-END-OF-JOB.	EDIT0001	
167	009080	DISPLAY 'END-OF-JOB' UPON CONSOLE.	EDIT0001	
168	009090	CLOSE CARD-IN, DISK-OUT, LISTING.	EDIT0001	
169	009100	STOP RUN.	EDIT0001	

Figure 6. COBOL Source Statements, 3 of 3

About Computer Sciences Corporation

EPG-II represents a foremost opportunity for the Data Processing Executive to improve his department's responsiveness to senior management while increasing the efficiency level of the department's performance. The proprietary EPG-II System is one of several advanced products from CSC, a total support organization dedicated to serving industry and government by designing and implementing systems. EPG-II mirrors the professional experience and technical competence of the Nation's leading independent computer software firm, the first of such firms to be listed on the New York Stock Exchange.

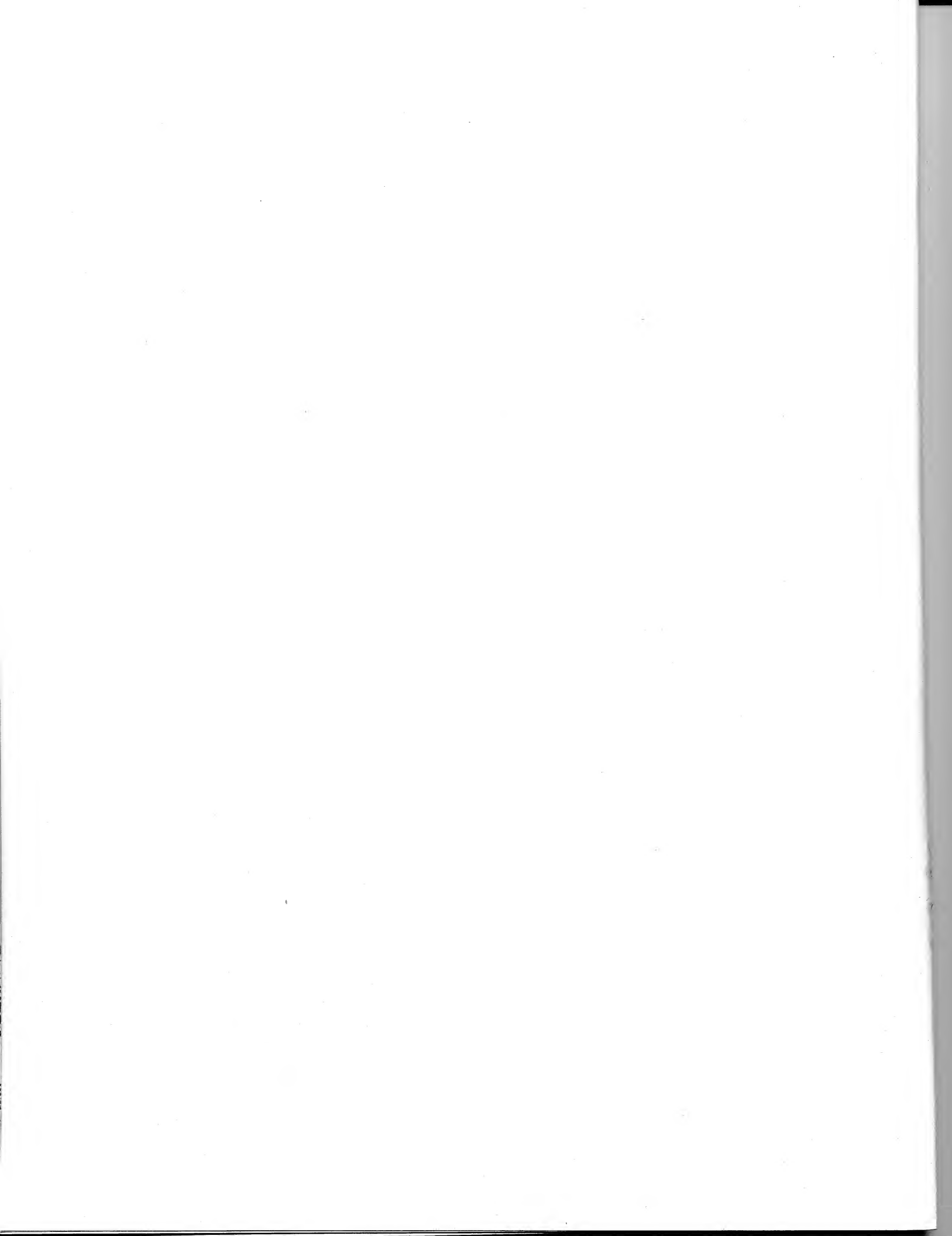
Since its founding in 1959, CSC has become a major source for the development and implementation of computer-based systems. CSC has performed systems analysis, design, programming, simulation, testing, implementation, time-sharing, and computer facility management services for many of the Nation's leading computer users. Its clientele includes a substantial number of the "Fortune 500" companies, every major manufacturer of computing hardware,

large civilian and military agencies of the Federal Government, and leading private institutions.

Without compromising the most rigid of standards, CSC has assembled one of the largest staffs of professional data system specialists in the world. This staff presents the highest level of experience and education among comparable organizations in the information sciences. Leading technically-oriented staff members are at the MA, MS, and PhD levels.

Seasoned management personnel include former presidents of leading corporations in the information sciences field, civilian governmental officials, senior military commanders, financial managers, manufacturing and transportation specialists, and managers of large computer-oriented complexes.

On any basis—technical superiority, efficient and prompt performance, number of working systems designed and programs completed, dollar volume of business, financial responsibility, or by other meaningful comparison—Computer Sciences Corporation leads in its field.



CSC

Applications System Division

COMPUTER SCIENCES CORPORATION

Offices and Facilities in Principal Cities Throughout the United States, Canada, Europe, and the Pacific Area